# CLAIMS

1. A method for partitioning container-managed state for a Java based application, comprising the operations of:

classifying individual entity bean objects with a particular state management type;

providing a plurality of state objects, each state object storing a state of a corresponding entity bean object within a memory address space of a Java server process, wherein each state object is associated with the state management type of the corresponding entity bean object; and

providing state management for each entity bean object based on the associated state management type using a corresponding state object.

2. A method as recited in claim 1, wherein the state management type is a memory replicated state management type.

3. A method as recited in claim 1, wherein the state management type is a disk replicated state management type.

4. A method as recited in claim 3, further comprising the operation of grouping the state objects based on the type of state management associated with the corresponding entity bean object.

5. A method as recited in claim 4, wherein the state management type identifies a policy for replication of a state object to a particular type of state server.

6. A method as recited in claim 4, wherein the state management type identifies a policy for migration of a state object from one server process to another server process.

7. A method as recited in claim 1, further comprising the operation of managing checkpoints using the state objects.

8. A method as recited in claim 1, further comprising the operation of performing lock management using the state objects.

9. A method for partitioning container-managed state for a Java application, comprising the operations of:

partitioning individual entity bean objects of the Java application into state partitions, wherein the state partitions manage concurrency for the Java application;

classifying individual entity bean objects within each state partition using state management units, wherein each state management unit identifies a particular state management mechanism for recoverable and migration capable state of corresponding entity bean objects.

10. A method as recited in claim 9, further comprising the operation of providing a plurality of state objects, each state object storing a state of a corresponding entity bean object within a memory address space of a Java server process, wherein each state object is associated with the state management unit of the corresponding entity bean object.

11. A method as recited in claim 10, wherein the state management units provide a mechanism to checkpoint the state objects to state servers.

12. A method as recited in claim 11, wherein a replica of each state management unit is maintained on the state servers.

13. A method as recited in claim 12, further comprising the operation of using a control module to manage dynamic partitioning of the state of the application via the state partitions and the state management units.

14. A method as recited in claim 13, wherein the state partitions and state management units are modular.

15. A method as recited in claim 14, wherein additional state management types for the state management units can be defined.

16. A method as recited in claim 15, wherein each state partition serializes transactions for entity bean objects within a particular state partition.

17. A method as recited in claim 16, wherein each state partition allows only one concurrent transaction to be performed on the entity bean objects within the particular state partition during a given time period.

18. A system for partitioning managed container-managed state for a Java based application, comprising:

an application having a plurality of entity bean objects;

a plurality of state objects, each state object storing a state of a corresponding entity bean object within a memory address space of a Java server process, wherein each state object is associated with a particular state management type; and

a plurality of state management units that classify the state objects based on the particular state management type associated with each state object, wherein the state management units facilitate state management for each entity bean object.

19. A system as recited in claim 18, wherein the entity bean objects of the application are partitioned into state partitions during pre-deployment.

20. A system as recited in claim 19, further comprising a repository that maintains state partition specifications for the state partitions.

21. A system as recited in claim 20, wherein the repository manages replication and migration of state of the Java application during runtime.

22. A system as recited in claim 18, wherein the state management type can be a memory replicated state management type.

23. A system as recited in claim 18, wherein the state management type can be a disk replicated state management type.

24. A system as recited in claim 18, wherein the state management type identifies a policy for replication of a state object to a particular type of state server.

25. A system as recited in claim 18, wherein the state management type identifies a policy for migration of a state object from one server process to another server process.